1993

An introduction to model integration and integrated modeling environments

Dolk, Daniel R.

North-Holland

Decision Support Systems, v. 10 1993, pp. 249-254

http://hdl.handle.net/10945/48243

Decision Support Systems 10 (1993) 249–254
North-Holland

An introduction to model integration and integrated modeling environments

Daniel R. Dolk

Naval Postgraduate School, Monterey CA, USA

Integrated modeling systems provide support for the definition, manipulation, and control of mathematical models throughout the entire modeling life cycle. Model integration is a particularly crucial operation which requires thinking about "modeling in the large", and which extends the scope of model management research to include manipulation as well as definition. Several aspects of model integration are identified and briefly described with respect to the problems they raise for constructing integrated modeling environments. Relevant work in these areas is cited. A brief introduction to each of the papers in this special issue is provided within the context established.

Keywords: Model integration; Structured modeling; Logic modeling; Graph-grammars; Integrated modeling environments

Daniel R. Dolk is Associate Professor of Information Systems in the Department of Administrative Sciences at the Naval Postgraduate School in Monterey, CA. Since receiving his Ph.D. in Management Information Systems from The University of Arizona in 1982, his research has focused primarily on model management and decision support systems. He has published extensively in this area in journals such as *Communications of the ACM, IEEE Transactions on Software Engineering, Interfaces*, and *Decision Support Systems*. He is currently Associate Editor for *ORSA Journal on Computing, Information Systems Research*, and *Journal of Database Administration*, and is a member of the ACM, IEEE Computer Society, and TIMS.

Correspondence to: Professor Daniel R. Dolk, Dept. of Administrative Sciences, Naval Postgraduate School (AS/DK) Monterey, CA 93943-5000, USA.

1. History

Since 1988, there has been an integrated modeling environments (IME) minitrack within the knowledge and decision support system track of the Hawaii International Conference on System Sciences (HICSS) devoted to model management research. The IME name was selected to convey the broadest spectrum of model management topics, and to encompass the diverse theoretical and practical issues related to the development of robust modeling environments for decision support.

Over the years, the HICSS IME sessions, in concert with their sister logic modeling minitrack sessions, have provided one of the premier forums for model management. Leading researchers have offered a rich mix of topics including model representation, visualization, graphical interfaces, integration, modeling languages, and model management system implementations. This special issue is, in some sense, a culmination of the excellent papers and discussions on model management which have characterized these IME sessions.

2. Model management and model representation

Model management is an interdisciplinary pursuit which combines elements of operations research (OR), artificial intelligence (AI), database management, management science (MS), cognitive science, and decision support (among others), each of which is an accepted discipline in its own right. This has the advantage of providing exciting research opportunities for people with interests in two or more of these areas. However, there is the tendency to coopt model management and

reduce it to the context of the corresponding reference discipline. Thus, for example, model management becomes conceptual modeling in the database community, knowledge base modeling in AI, and mathematical modeling in the OR/MS world.

Model management, as interpreted by the participants in the IME sessions, usually refers to the application domain of mathematical modeling, with particular emphasis on OR/MS models for decision support. This characterization is unnecessarily restrictive, however, for model management really entails much more than just this province of applications. For example, computer-aided software engineering (CASE) tools now speak routinely of model management in the sense of storing, retrieving, and manipulating the conceptual data models and process models which comprise a specific information architecture.

Modeling is a pervasive activity which manifests itself in nearly every discipline. In the information systems world, for example, one can argue that it is the models that are of primary interest rather than the data which historically has been the focal point. We do not understand an organization by looking at what data it has accumulated but rather by what models it operates under, for example its models for risk analysis, financial accounting, and decision making in general. Although mathematical modeling is the starting point for most of model management research, one cannot help but be struck by the similarities across modeling disciplines. The potential significance of research findings in model management extends well beyond the sphere of OR/MS or any other patently model-based field.

During the last decade, the major advances in model management have been in the area of representation. Three main schools of model representation have emerged from this research: Structured modeling, logic modeling, and graph grammars. Structured modeling [11] is a formal definitional scheme based upon graph theory which extends semantic data models from the database world to capture the complexities of mathematical modeling. Structured modeling allows the user to view models graphically, textually, or algebraically, and at different levels of abstraction. Logic modeling [14] is more a wedding of artificial intelligence with mathematical modeling, relying primarily upon first order logic as the means of capturing model knowledge. The allure of this approach is that first-order logic provides a very powerful, formal means of description which encompasses a ready-made executable language in the bargain. Graph grammars [13] offer a flexible visual metaphor of model representation which can best be stated as "models as graphs". This graphical paradigm for describing and manipulating models is particularly conducive to implementation, given the recent emergence of powerful, inexpensive graphics software.

These model representation schemes are neither mutually exclusive nor collectively exhaustive. There is a significant area of overlap among them and attempts to integrate the schemes have been made. Structured modeling, logic modeling, and graph grammars are well represented in this special issue, and several papers explicitly treat the cross-play between them.

3. Model integration: beyond model representation

One of the objectives of the IME minitrack has been to go beyond strictly representational issues to consider how models may be linked, or integrated, with one another. The motivation for this focus reaches back to the relational data model from database research. The relational model includes not only a representation dimension (data as tables) but a powerful manipulation formalism as well (relational algebra/predicate calculus with transitivity). This closed world property of the relational model provides a compact elegance which eludes the more complex domain of model management.

To demonstrate the difficulties inherent in model integration, consider a situation where we have a standard transportation model for which the demand and supply components are determined from other independent models. For example, demand may be forecasted from a maximum likelihood estimation model and the supply component may be determined from a discrete event simulation manufacturing model (see fig. 1). When we speak of integrating these simple models, the following kinds of integration must be considered:

(1) schema; (2) process; (3) models and data;

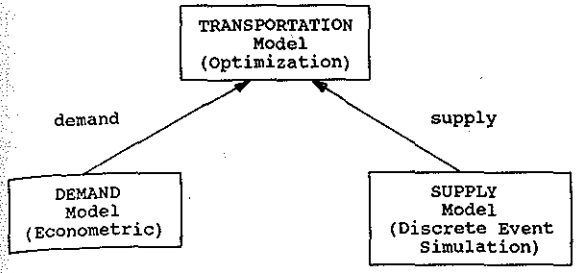

Fig. 1. Integrated multi-paradigm model.

(4) models and solvers; (5) modeling paradigms; (6) environments; (7) modeling system interfaces. Although this is by no means an exhaustive list, these aspects of integration raise very challenging research issues.

Schema integration is a problem which has been studied extensively by the database community [1] and which has direct corollaries for model management. Geoffrion [12], for example, suggests a methodology for integrating structured modeling genus graphs to form composite models at the logical, or definitional, level. Although at first blush, simple splicing (i.e., 'joining' graphs at points they have in common such as *demand* and *supply* in fig. 1) may seem sufficient to create an integrated model, in general the problem of where and how to 'join' independent genus graphs and their associated schemas is non-trivial and involves complex issues of inheritance and abstract data types (see, e.g., [3,2]). Further, it seems that, even using a formal system such as structured modeling for representation, integration at the schema level is very difficult to automate completely; some human adjudication is required along the way for all but the simplest of cases.

An integrated structured model schema can be seen as the logical, or representation, dimension of integration but there still remains the question of how to manipulate this integrated model. Assuming model solution as the manipulation operator, there are several ways this might occur. If each model component has a separate solver, then one scenario might be to simply 'pipeline' solvers, that is, to link them serially so that outputs of any one model feed directly to the inputs of its successor. This technique could be used for our integrated distribution example, however, there are instances where it might not always be appropriate. Combining several regional transportation models into an integrated national transportation model, for example, could be viewed as a consolidated integration where the same solver that is used for each of the component models is also used for the integrated model.

In the general case, it's not clear from simply viewing the logical structures of component models and their integration what processes must be invoked to solve the integrated model. A process, or solver, integration schema in the form of an MML which complements a model description language is necessary as well. Muhanna and Pick [19] describe a systems-oriented approach to this problem with a graphical model integration interface whereas [15] presents features of an MML based on the notions of communicating sequential processes. Object-oriented approaches for process integration have also been elaborated ([5,18]). Research is just beginning to focus on this aspect of modeling languages after nearly a decade of concentrating on model description.

One of the biggest obstacles in the development of complex mathematical models is obtaining valid data to populate a meaningful model instance. The integration of models and data often entails the lion's share of a large-scale modeling project. It is essential that modeling systems have access to existing corporate data resources to minimize this impact. This raises the issue of how to integrate data management technology such as query languages [4] and dictionary, or repository, systems [7] into IMEs. Relational database systems do not seem powerful enough to support mathematical modeling in general and the advent of extensible and object-oriented DBMS as implementation vehicles for IMEs is a fertile area for research [6].

A staple of OR research since its inception has been the development of efficient algorithms for solving classes of optimization models. Model management has attempted to expand this narrow solver-oriented approach to modeling by searching for more robust representations such as those described above. Solvers are nevertheless critical elements in an IME and it's necessary for an IME to be able to integrate high level model representations with the specific data structures required by solution algorithms. Research is required about how to do this without having to write a specific transformation for every combination of representation and solver [17]. The paper

by Ramirez, Ching, and St. Louis in this issue addresses the problem of model/data/solver independence in detail.

Model representation coupled with data and solver integration gives rise to the notion of modeling paradigm integration. This addresses the ambitious objective of being able to support a wide range of modeling paradigms (e.g., mathematical programming, discrete event simulation, econometrics, database, etc.) within a single, uniform environment. Realization of this objective depends not only on the robustness of the underlying model representation but also on the ability to link a wide range of corresponding solvers such as simulation programs, statistical routines, and database query processors. [16] and [8], for example, discuss the problems inherent in trying to adapt the relatively static formalism of structured modeling to the representation of dynamic behavior as embodied in discrete event simulation models. Moving the frontiers of model management beyond mathematical programming where most of the successes have been enjoyed is one of the more interesting research challenges facing us today. Only when diverse paradigms can be represented under a single umbrella will it become truly efficient to consider model integration in the broadest sense.

Another area of integration that is becoming increasingly important, not just for model management but for all applications is software integration. The ability to link word processors, spreadsheets, databases, and graphics packages in a seamless fashion enhances the feasibility of modeling environments which can support paradigm integration. Contemporary operating systems are providing increased capabilities in this vein through such features as dynamic data exchange (DDE).

Finally, an IME must be able to support interfaces for other familiar modeling systems. In the same way that command-driven operating systems can be tailored to look like others (e.g., Unix to look like DOS) by renaming commands, an IME should be able to support other modeling system interfaces so that users can work in a venue in which they are comfortable. An optimization modeling environment, for example, might facilitate a GAMS or LINDO interface for users familiar with those systems. This would obviate them having to learn a new environment. The challenge in this arena of integration is to construct efficient transformations from these external system representations to the internal model representation underlying the IME. The Bhargava and Kimbrough paper in this issue offers novel approach to solving this problem.

4. Summary of papers

The inspiration for this special issue arose largely from an especially productive and high caliber IME session in 1990. [1] Although preliminary versions appeared in the 1990 HICSS Proceedings, the articles in this special issue have undergone extensive review and subsequent revision from their HICSS counterparts to ensure they meet journal publication criteria.

One of the premier thrusts in model management has been the search for more effective languages to support the modeling process across multiple paradigms. The papers in this issue reflect this emphasis faithfully.

Chris Jones' "An integrated modeling environment based on attributed graphs and graph grammars" describes *networks*, a graph-based modeling system (GBMS) for representing and manipulating models, and shows how it facilitates model integration. Four specific models are constructed and subsequently linked: Critical path, linear programming, network flow, and decision tree. The striking feature about *networks* is that everything is done graphically, even the operations on the graphs of the models themselves. This visual approach constitutes a dramatic departure from traditional model building which has tended to rely heavily upon the algebraic representation of models. In many cases, the graphical medium appears to offer an intuitively appealing way for users to view models; network flow models, for example, certainly lend themselves exceptionally well to this representation.

Jones also provides a concise overview of model integration research, showing how graph-grammars compare with structured modeling and logic modeling in this context. This survey indicates many of the overlaps between these three ap-

[1] Chari and Krishnan's paper was presented in the 1990 Logic Modeling session.

proaches to model description, as well as providing a convenient transition to the next two papers in this special issue.

Bhargava and Kimbrough's article, "Model management: An embedded languages approach", is representative of the logic modeling school of model management. The authors describe a technique they call *embedded languages* which lends powerful support for the modeling system interface brand of integration described above. The basic idea is that an executable embedding language can be built (and has been built for a Coast Guard application) which can contain one or more embedded languages, including existing executable modeling languages (EML). The embedding and embedded languages are represented formally in first order logic, and constitute elements of an overall target language which sits on top of them. Incorporating external languages as embedded languages allows an underlying modeling environment to accrue functionality in an incremental fashion according to the strengths and weaknesses of specific EMLs. Thus, if we wish to construct a modeling environment to support the integrated model example discussed above, we might embed EMLs for linear programming (e.g., AMPL [10]), econometric modeling (e.g., PERM [9]), and discrete event simulation (e.g., Simscript).

Embedded languages offer a viable means for building bridges between the three paradigms for model representation. Chari and Krishnan's paper, "Towards a logical reconstruction of structured modeling" provides a blueprint for embedding structured modeling within logic modeling. The authors show how to recast structured modeling into an embedded language in first order logic thus providing a concrete example of how the embedding process can be implemented.

Whereas embedded languages constitutes an incremental approach to developing a language for modeling systems, Hong, Mannino, and Greenberg propose a universal modeling language in "Measurement theoretic representation of large, diverse model bases: The unified modeling language ℓ_U". ℓ_U is a declarative language based on logic modeling and measurement theory. Measurement theory is concerned with mapping the empirical world of objects and relations into mathematical systems, and provides a very relevant, though previously unexplored, theoretical basis for modeling languages. ℓ_U is capable of representing very broad kinds of knowledge about models including assumptions, and subsumes much of the functionality of object-oriented languages.

Ramirez, Ching, and St. Louis' paper, "Independence and mappings in model-based decision support systems", is also about modeling languages, but focuses on an actual implementation rather than on theoretical concerns. The authors discuss the issues surrounding model, data, and solver independence, and then describe the data and algebraic model system (DAMS) which supports these levels of independence. DAMS is an implementation of a modeling environment based on structured modeling which contains an SM/DB modeling language as well as an extended version of SQL, called SQL/OBJ, for data manipulation.

5. Conclusions

One of the implications of the articles in this issue is the level of difficulty involved in building an IME. Many fine modeling systems exist for specific applications such as linear programming and statistical analysis, but systems which support a wide range of modeling paradigms are virtually non-existent. The inability to share models across systems designed for specific domains artificially restricts the utility and impact of modeling in general. Continuing to rely upon standalone systems segmented for particular applications will ensure that the modeling community remains fragmented and less effective than it can be.

The hope of researchers in model management is to reveal the coherence of the modeling process independent of discipline, and to show that environments can be developed which support effectively these more general aspects of modeling. Only by achieving this high level of integration are we likely to see the full realization of modeling as an indispensable part of organizational decision making.

6. References

[1] C. Batini, M. Lenzerini, and S.B. Navathe, A comparative analysis of methodologies for database schema integration, ACM Computing Surveys 18, 4 (December 1986) 323–364.

[2] H. Bhargava, S. Kimbrough and R. Krishnan, Unique names violations: A problem for model integration, ORSA Journal on Computing 3, 2 (1991) 107–120.
[3] G.H. Bradley, and R.D. Clemence, Jr., A type calculus for executable modeling languages, IMA Journal of Mathematics in Management 1, 4 (1987) 277–291.
[4] J. Choobineh, SQLMP: A data sublanguage for representation and formulation of linear mathematical models, ORSA Journal of Computing 3, 4 (Fall 1991) 358–375.
[5] M.A.H. Dempster, and A.M. Ireland, Object-oriented model integration in a financial decision support systems, Decision Support Systems 7, 4 (1991) 329–340.
[6] S. Desai, Are extensible database systems better than relational database systems for model management?, Anderson Graduate School of Management, UCLA, Los Angeles, CA, 1991.
[7] D.R. Dolk, Model management and structured modeling: The role of an information resource dictionary system, Communications of the ACM 31, 6 (June 1988) 704–718.
[8] D.R. Dolk, Structured modeling and discrete event simulation. NPS Working Paper No. 90-03, Department of Administrative Sciences, Monterey, CA, August 1990.
[9] D.R. Dolk, and D.J. Kridel, An active modeling system for econometric analysis, Decision Support Systems 7, 4 (1991) 315–328.
[10] R. Fourer, D. Gay and B.W. Kernighan, A mathematical programming language, Management Science 36, 5 (May 1990).
[11] A.M. Geoffrion, Introduction to structured modeling, Management Science 33, 5 (May 1987) 547–588.
[12] A.M. Geoffrion, Reusing structured models via model integration, Proceedings of the Twenty-Second Annual Hawaii International Conference on System Sciences, Vol. III (IEEE Computer Society 1989) 601–611.
[13] C.V. Jones, An introduction to graph-based modeling systems, Part I: Overview, ORSA Journal on Computing 2, 2 (1990) 136–151.
[14] S.O. Kimbrough and R.M. Lee, Logic modeling: A tool for management science, Decision Support Systems 4, 1 (April 1988) 3–16.
[15] J.E. Kottemann and D.R. Dolk, Model integration and modeling languages: A process perspective, Information Systems Research 3, 1 (1992) 1–16.
[16] M. Lenard, Extending the structured modeling framework for discrete-event simulation, Proceedings of the 25th Hawaii International Conference on System Sciences, Vol. III (IEEE Computer Society Press, January 1992).
[17] S. Maturana, Integration of a mathematical programming solver into a modeling environment, Anderson Graduate School of Management, UCLA, Los Angeles, CA, October 1988.
[18] W.A. Muhanna, An object-oriented framework for model management and DSS development, Forthcoming in Decision Support Systems.
[19] W.A. Muhanna and R.A. Pick, Composite models in SYMMS, Proceedings of the 21st Hawaii International Conference on Systems Sciences, Vol III (IEEE Computer Society Press, January 1988) 418–427.